جـــامـعــة أم الـــقـــرى

UMM AL-QURA UNIVERSITY

Propagation Delay Acceleration in Blockchain Network

Thesis Submitted to

College of Computer Science and Information System of

Umm AlQura University In partial fulfillment of the requirements for

The Degree of Master of Science

BY

Rayan Saeed Alsulamy

43780300

UMM ALQURA UNIVERSITY

SUPERVISED BY

Dr Khalid Altarmisi

Academic Year 1441/2020

Acknowledgments

I would like to express sincere gratitude to my major advisor Doctor Khalid Altarmisi. I am deeply indebted to him for his guidance on my research and my course works. The door of his office was always open whenever I ran into a spot of trouble or had a question about my research or writing. He was very patient with me and always encouraged me during the whole process. I would like also to acknowledge the continuous support of the XDean of College of Computer and Information Systems (CIS) at Umm Al-Qura University (Dr. Fahad Al-Dosari), the Vice XDean of CIS for Research and Graduate Studies (Dr. Waleed Al-Asmari), and all faculty members and staff in the CIS.

Dedication

This thesis is dedicated to my parents who always support and help me. This work is also dedicated to my family, who have been a source of support and encouragement during the challenges of graduate school and life. This work is also dedicated to my supervisor and my friends.

Abstract

Blockchain is a new revolutionary technology that was essentially developed to eliminate centralized authority on the internet. Since its inception, it has expanded rapidly, and new decentralized applications and currencies have been developed. Its security makes transactions immutable and more trustworthy in an environment of anonymity which combine three main technologies. First concept of public key infrastructure and digital signatures for proofing the ownership of transactions within a public network. Second peer 2 peer for a direct connection between the participants without third party. Lastly consensus protocol in which making sure to reach an agreement among most of the participants in the network. However, it still has many vulnerabilities and issues that must be studied and addressed. One of the challenges of blockchain that brings a lot of security issues is the delay in the propagation among the blockchain network. In this thesis, a new method will be proposed to enhance the delay propagation and specifically to minimize transaction verification by using Different Digital Signature Cryptosystem.

ملخص الرسالة

سلسلة الكتل هي تقنية ثورية جديدة تم تطويرها بشكل أساسي للتخلص من أي سلطة مركزية على الشبكة العنكبوتية منذ نشأتها اكتسبت توسعًا سريعًا وتم بواسطتها تطوير تطبيقات لا مركزية جديدة وعملات رقمية. بسبب أمانها الذي يجعل المعاملات غير قابلة للتغيير وأكثر ثقة في بيئة تحافظ على سرية الهوية والتي تجمع بين ثلاث تقنيات رئيسية. المفهوم الأول البنية التحتية للمفتاح العام والتوقيعات الرقمية لإثبات ملكية المعاملات داخل شبكة عامة. تقنية النظير للنظير للاتصال المباشر بين المشاركين دون طرف ثالث. وأخيرًا بروتوكول توافقي يتم فيه التأكد من التوصل إلى اتفاق بين معظم المشاركين في الشبكة.. ولكن مع ذلك ، لا يزال لديها العديد من نقاط الضعف والقضايا التي يجب دراستها ومعالجتها. أحد تحديات سلسلة الكتل التي تسبب الكثير من المشكلات الأمنية هو التأخير في الانتشار بين شبكة سلسلة الكتل. في هذه الرسالة ، سيتم اقتراح طريقة جديدة لتعزيز انتشار التأخير وتقليل التحقق من المعاملات على وجه التحديد باستخدام نظام تشفير و توقيع رقمي مختلف.

Table of Contents

List of tables

List of abbreviation

ECC	Elliptic curve Cryptography
NTRU	N-Th Degree Truncated Polynomial Ring
ECDSA	Elliptic curve digital signature Algorithm
NTRUSign	NTRU digital signature
POW	proof of work protocol
POS	proof of stake protocol
TLS	Transport Layer Security
PGP	Pretty Good Privacy
SSH	Secure Shell protocol

List of figures

Chapter 1 Introduction

Brief history

For any process of buying and selling commodities through the internet, there is a need for a trusted third party to deal with to complete the process of buying and selling. As a consequence, there is a need to trust those third parties and provide them with private information only for verification processes while exposing most information is not necessary. As a result, there was an attempt to protect users' privacy in the late 1980s under the name of CypherPunk. The goal of this act is to protect the privacy and security of people's information by using cryptography technologies. As Erick Hughes, a CypherPunk activist posted in 1993, "in the electronics age" people should have a choice to reveal their private information to whom they want and hide it from whom they want in the network. Indeed, this idea spread widely even though it was against laws causing large illegitimate internet actions [1]. The first implementation of this concept was in 1999 by a music sharing app called Napster which was founded by Sean Parker and Shawn Fanning with a protocol known as peer 2 peer protocol. A lot of music was shared illegally by using Napster.

Another sharing file system called BitTorrent was founded by Bram Cohen in 2001. BitTorrent works by using BitTorrent Client for connecting with other clients. Swarms in BitTorrent exchange by requesting pieces of files (download) and send files that were requested (uploading). But with all its popularity, it still has one vital weakness, which is that the client's IP address can very easily be exposed.

In addition to the previous two attempts, there were many other attempts that tried to share data through the network. The two closest to what is known as Blockchain are B-money and Bit-Gold.

B-money was published in 1998 by Wei Dai. This publication was the first distributed system that dealt with hash cash. Wei described two protocols of which the first one is not practical [2]. In the second protocol, every one of the participants is known in the network by just a public key. The public key acts as an

ID for each one in the network. All the participants in the network are known by maintaining a database that contains all participants' IDs, which is distributed separately. The creation of money happened by solving computational problems and the amount is determined by the difficulty of the problem. The transferring of money in B-money had similar functionalities that exist nowadays in common crypto currencies. Wei used a consensus protocol in which each party of the network must approve the transaction. He used also digital signatures to verify the identity of the sender and receiver. Unfortunately, his protocols remained as a proposal and were never applied in a real environment. Nevertheless, Satoshi referenced Wei Dai's proposal in his paper as one of the inspirations for Bitcoin

The second idea was Bit-Gold. A crypto system that appeared in 1998 by Nick Szabo with also the same idea as Bitcoin. However, it was not applied like B-money. Some researchers said it was a precursor of the bitcoin because it was based on the Proof of Work (POW) protocol and used a Byzantine Fault Tolerance peer to peer network. Differing from Bitcoin, Bit-Gold used a method that depends on a quorum of addresses that was vulnerable to Sybil attack[3].

What is Blockchain

Blockchain technology is a general name of an intelligent idea that emerged in 2008 by an unknown person named Satoshi Nakamoto[4]. The main concept of the Blockchain is building a trust-based decentralized network and eliminating any rules of centralized authority inside it.

The word Blockchain contains two parts. A block, which refers to the container that holds all the data in the network during a period of time and chain which refers to the process of stacking each block to the previous one to perform a chain. Blockchain is based on a decentralized peer 2 peer system which adopts ledgers to keep a record of all transactions that took place inside the network. The main properties of Blockchain are transparency and immutability, where transparency means that each node inside the blockchain network has a copy of the same data shared over the network, which means that all the data is shared transparently. For this reason, the data is immutable so changing or tampering it is nearly impossible.

Within a specific timeframe, all the transactions will be contained in a single block. An algorithm is initiated to find this block and then that block will be stacked with other blocks to create a chain.

All the nodes in the blockchain network also act as witnesses as a result of having a copy of all the information of events inside the network which makes block tampering nearly impossible. For this reason, utilizing of blockchain technology evolved very quickly after initiating the first blockchain application[5].

Blockchain applications

In the early days of Blockchain, it was not known to the public and it was not until digital currencies appeared, that use blockchain technology came to the surface. One of the most popular applications of blockchain is Bitcoin.

Bitcoin

The author Satoshi defined Bitcoin in his white paper as “A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution.”[1] The valuable digital unit in the Bitcoin is called a Bitcoin. Users can use the digital Bitcoin unit to buy and sell assets, transfer money, and exchange bitcoin with other cryptocurrencies.

Ethereum

In 2015, Vitlaik Buterin launched Ethereum which is the first programmable digital currency. The digital unit of the Ethereum is called ETH. It is also completely decentralized. Ethereum spreads quickly because it uses a smart contract. The core difference between Bitcoin and Ethereum is that Ethereum is based on a smart contract that the programmer can build in an application to accomplish a user’s demands, and the application can be used in a decentralized network. Ethereum has many applications now like financial applications, games and decentralized market applications[6][7].

Besides these two applications of Blockchain, Blockchain technology is not restricted to only cryptocurrency, it also used in economics, medicine, software engineering, and the internet of things.

Components of Blockchain

Blockchain stands for three parts.

Cryptography, which uses ECC (Elliptic Curve Cryptosystems) that is based on PKI (Public Key Infrastructure). It uses two public and private keys. A public key is represented as a locker for data while the private key is represented as a key that can open the lock. In addition, Blockchain uses a specific algorithm in ECC which is called Elliptic Curve Digital Signature Algorithm (ECDSA). This algorithm helps in proofing the user's ownership of the signature without exposing the private key. ECDSA works by using the private key to encrypt a message and the public key to verify the signed message as encrypted from that private key (see Figure 1).

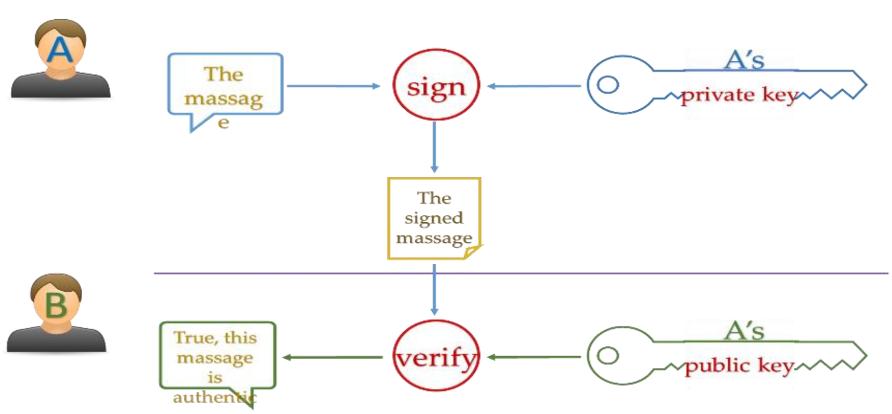

Figure 1 Sign and verify a message by using a digital signature

Peer 2 peer. This concept is based on eliminating the role of third parties for providing the process of verification and building trust. As a result, each participant in the network can connect and share data directly with other participants by following specific rules that were found to ensure trust amongst all of them. Figure 2.

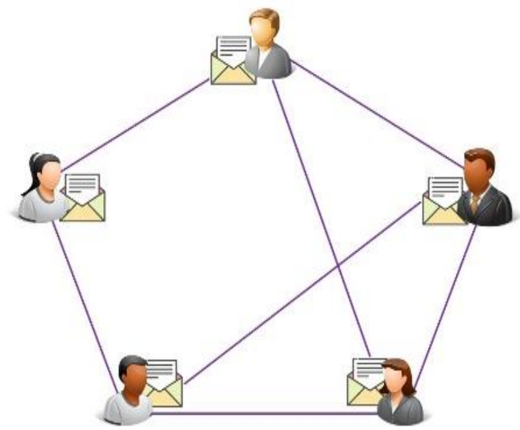

Figure 2 Peer 2 peer network

Game theory A consensus algorithm's goal is to reach an agreement amongst most of the nodes in the network for every process confirmation. The two most popular types of consensus protocols are Proof of Work (POW) and Proof of Stake (POS) that are used in Bitcoin and Ethereum respectively. POW works by solving hash puzzles that need huge computational power to find a hash with a SHA256 algorithm which meets difficulty requirements. On the other hand, POS works by locating the nodes with a higher amount of currency. According to POS, the one who owns more money in POS has less chance to be an attacker and a higher chance to be a new block initiator. In addition, there are many other protocols used in digital currencies. All those protocols work to ensure that all nodes are in a consensus statement.

Blockchain Structure

As previously mentioned, Blockchain is based on a decentralized network that does not need a third party. All nodes work as a distributed ledger. Blockchain contains chains of blocks and every block can be known by a hash algorithm in its header. Each block header also refers to a previous block or a parent block until reaching the genesis block which is the first block in the chain (see Figure 3) [7].

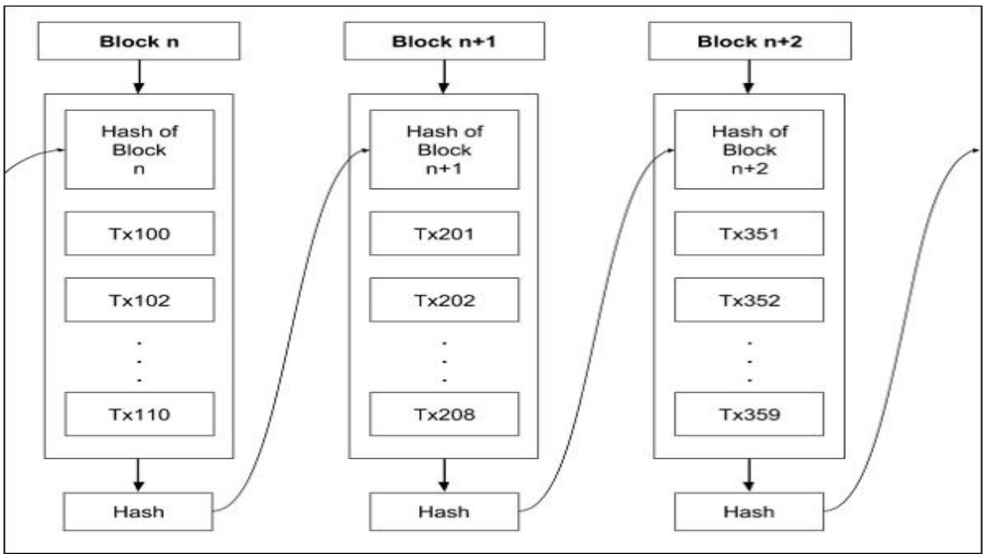

Figure 3 Chain of blocks representing each hash depends on previous hashes

Each block performs a number of transactions that contain the processes for transferring services from one node to another node. In addition, the transactions in the blocks will be performed in a tree called a binary tree (see Figure 4). Every two transactions will be hashed in one parent until it ends up with only one root hash called the Merkle tree which is used to provide immutability and integrity.

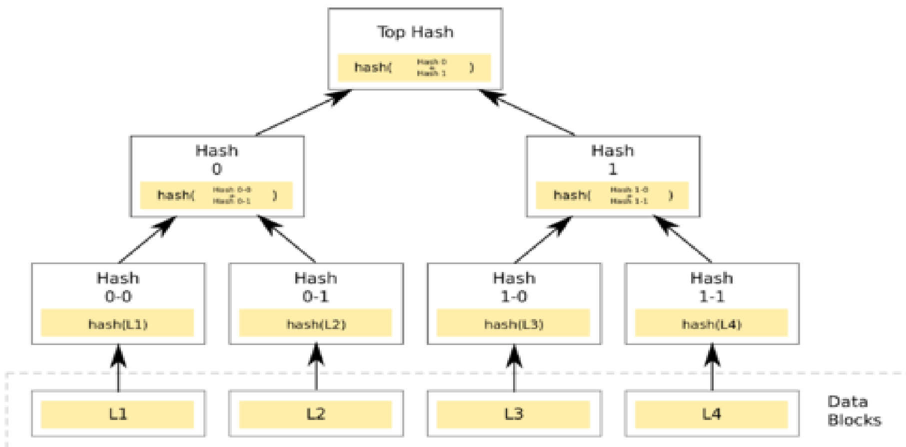

Figure 4 Binary Tree

Chapter 2 Bitcoin network

One of the most popular examples of Blockchain is Bitcoin. In this section, Bitcoin will be explained step by step.

Structure

The structure of Bitcoin is based on a P2P network which is represented as a number of nodes that are connected with each other directly. There are several types of nodes in the blockchain.

The first type of nodes is miners. In a Bitcoin network, only 10 percent of the nodes work as miners which are responsible for hashing and putting the transactions inside a block every 10 minutes. Meanwhile, the rest of the nodes are divided into two types.

The second type is full nodes that obtain a full copy of all the blockchain data in their local storage.

The last type is lightweight nodes. They do not obtain the full blockchain database, but they can still verify and propagate transactions in the network. For both propagation and verification, Bitcoin uses Elliptic Curve Cryptosystems and the hash function.

How Bitcoin works

Creating an address

In Bitcoin, the wallet software is responsible for generating the public and private keys. After initiating a new wallet, the wallet will start to generate an anonymous random number with the size 2^{256} by using the SHA256 hash algorithm as a private key. After that, it generates a public key by using Elliptic Curve Cryptosystem $K_{Pub} = K_{priv} * G$ in which G is a constant point called the generator point. Finding the private key from the public key is nearly impossible because it's a one-way process.

After having the public key, the wallet will derive the address from the public key. The address of the node is represented by a number beginning with 1 and it's necessary for sending and receiving bitcoin. The address is a result of the public key hashed with SHA256 then hashed with Race Integrity Primitive Evaluation Message Digest (RIPEMD) K_{Pub} hash = RIPEMD(SHA256(K_{Pub})[7]. Finally encoding the K_{Pub} hash with CheckSum58 to make a readable address. CheckSum58 contains all the numbers plus the alphabet but [I , l , 0 , O] are omitted to avoid ambiguity.

Create transactions

Transactions are the main container for transferring data from one address to another address inside a Bitcoin network. Transactions consist of inputs and outputs. An input is an amount of money that the sender wants to send to the receiver. An output is the amount of money that the receiver will get. The output as it is shown in Figure 5 is usually slightly less than the input because of the fees which the miners will collect for inserting the transaction in a block. The transaction also must be signed with the sender's private key and include his public key inside the transaction for his ownership to be verified. The transaction input is derived from previous output, and from one input more than one output can be initialized. For example, let's assume that Alice, who has an established address, wants to send Bitcoin to Bob's address. Alice will create a transaction that describes the amount of Bitcoin which will be sent to Bob's address as an input which is derived from the previous output from Joe (see Figure 5). Alice will sign the transaction with her private key and propagate it to the network [1], [7].

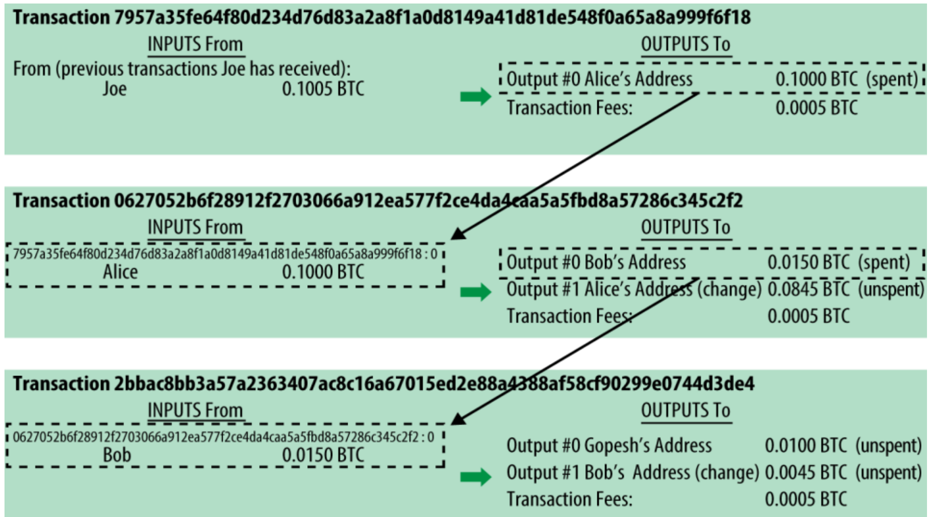

Figure 5 Alice using the output value from Joe as an input value to Bob [7]

Finally, all the nodes in the network will verify the transaction with the time they receive it by using Alice's public key which is embedded in the transaction. After Bob receives the transaction, the funds will be unconfirmed until it is embedded inside a block. Bob waits 10 minutes until the process of proof of work protocol, which will be explained later, is completed in order for the funds to be embedded inside the block.

Propagation mechanism

The Blockchain propagation mechanism differs from application to application. One of the most common examples of such a mechanism is the Gossip protocol. Bitcoin uses the Gossip protocol to propagate a transaction in the network. For Example, if Node 1 has a transaction to send then Node 1 will send an *inv* message to its peer Node 2 to check if Node 2 has seen the transaction before or not. If Node 2 does not have the transaction in its transactions list, then Node 2 will send getData to Node 1 to fetch the full transaction (see Figure 6).

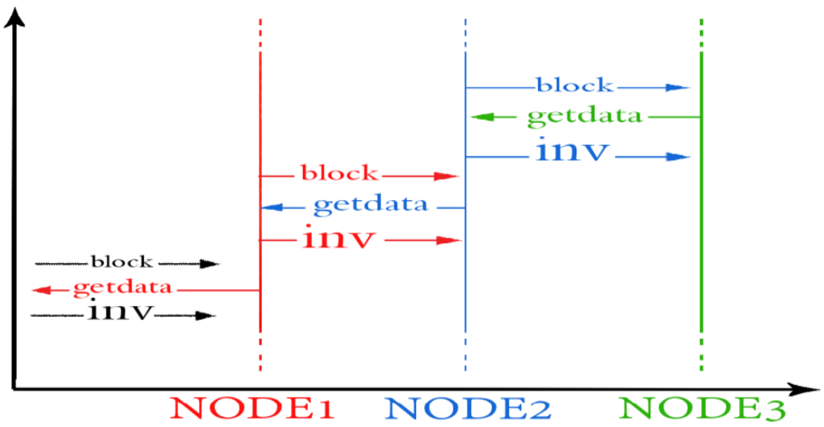

Figure 6 Exchange transactions and Blocks between the nodes in Bitcoin network

Proof of work

Proof of work is a bitcoin algorithm constructed to reach consensus among miners and determine which miner among thousands of miners could establish the block. All the miners must brute force a mathematical puzzle until reaching the predetermined hash. If the result was founded by one of the miners first, he has a priority to establish a block and put all the transactions in the last 10 minutes and propagate the block. If the block is confirmed by 51% of miners, then the block will be valid, and the miner will get a reward. The difficulty of the hashing puzzle increases based on the number of blocks per hour[7][8]. Furthermore, the algorithm voting system is based on the computational power of the node instead of the IP.

Bitcoin Technical Challenges

Even though Bitcoin was designed to solve original network obstacles like single failure and single authority, it still is suffering from some issues and needs more time to overcome its problems. For example:

Scalability

Every day most of the Bitcoin network is growing quickly, as a result, it is making propagation time in the network slower.

Usability

In Bitcoin, miners compete for Bitcoin rewards by solving a mathematical puzzle that takes 10 minutes to find a solution and that needs high energy-consuming power hardware like GPUs and special servers.

Throughput

Services like visa can do 2000 transactions per second, however, Bitcoin needs 10 minutes for each transaction's confirmation.

Bitcoin issues

Even though Bitcoin is secure, it still suffering some risk issues. For example:

Majority attacks

In Bitcoin, an attacker who acquires 51% of the hash power of the network has the ability to confirm and reject any transactions and blocks in the network. In 2014 a mining pool called Gash.io possessed 42% of hash power in bitcoin (see Figure 7) [9].

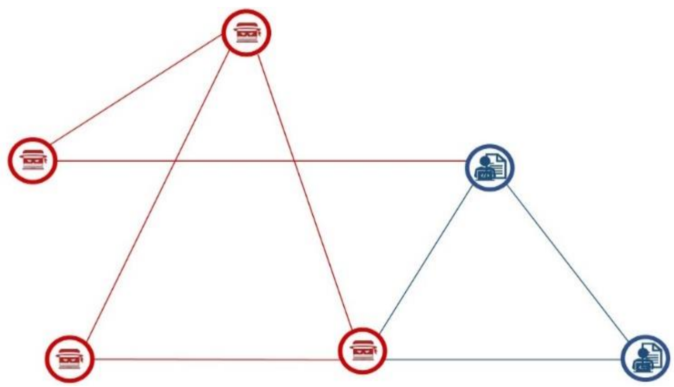

Figure 7 Majority attack

Double spending attack

An attacker can spend the same coins in two transactions. This attack can be applied by exploiting the time between the initial transaction and confirmation time in the first transaction and propagate the second transaction (see Figure 8).

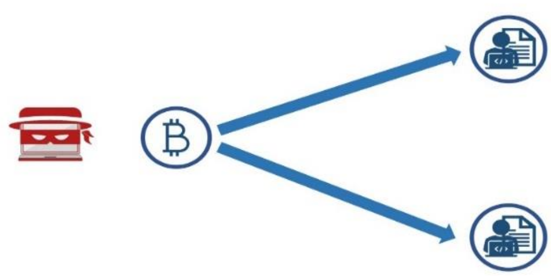

Figure 8 Double spending attack

Inconsistency

While the Bitcoin network is growing, inconsistencies between nodes are getting harder, which will increase the risk of issues like double spending, forking and eclipse attacks.

Forking

Forking is the most common problem related to the delay in propagation which occurs when two nodes find the result and propagate two blocks nearly at the same time. For this reason, the blockchain will be separated into two paths and each path will be mined separately by a group of nodes. The chain that becomes longer and has more block associated with it, eventually will be authorized if the nodes which have the shorter chain of blocks were notified about the longer one. The shorter one will be forged and it is called orphan blocks (see Figure 9).

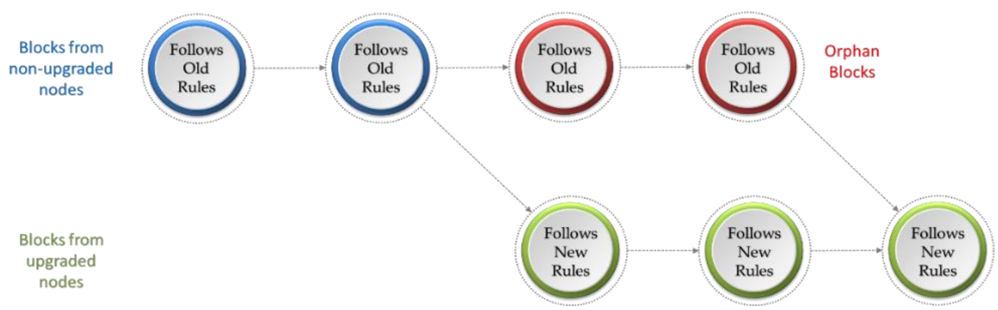

Figure 9 Two miners established two blocks at nearly the same time

Eclipse attack

Eclipse attacks happen when the attacker controls all the outgoing and ingoing connections for a specific node and can isolate the node from the network (see Figure 10).

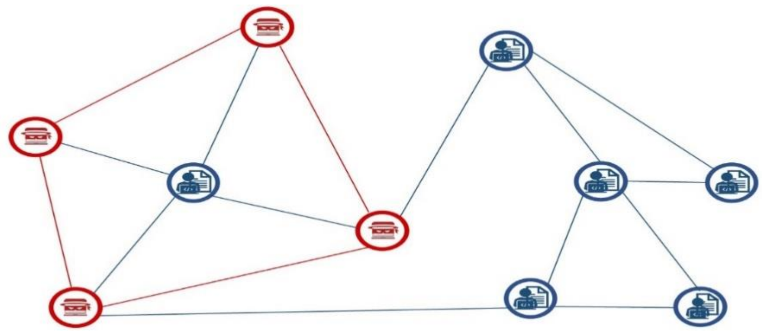

Figure 10 Eclipse attack

Problem statement

One of the most important research problems in Blockchain is the propagation delay. The propagation delay is combination of two parts transmission time and verification time. The verification time is the time is taken to verify a block or a transaction while need accessing to stored disc [10].

The importance of this research problem is that it causes some vulnerabilities such as fork and double spending attacks, in which they must be considered.

Contribution

Based on our studies, we classified the enhancement propagation delay solutions into four categories:

1. Change consensus protocol
2. Minimize verification time
3. Propagation protocol
4. Network topology

After reviewing previous works, a new method for propagation time will be proposed to reduce the verification time by using NTRU cryptosystem digital signature instead of ECDSA, and as a result, it will minimize the propagation delay.

Organization

This thesis is organized as follows. Chapter 1 introduced blockchain and Bitcoin. Chapter 2 explains ECC cryptosystem in more detail. Chapter 3 offers an explanation of NTRU cryptosystem. Then, chapter 4 compares NTRU and ECC in encryption and digital signature. Lastly, chapter 5 which contains details about how to accelerate propagation delay by using the NTRUSign verification process.

Chapter 3: Blockchain cryptography (ECC)

Introduction

ECC is an abbreviation of Elliptic curve cryptography. Which is one of the most popular asymmetric cryptosystems. It was conducted by Neal Koblitz and Victor S. Miller in 1985 and started to be widely used from 2004 to 2005. ECC is adopted by a number of famous technologies like TLS, PGP and SSH. In 2008, after using ECC in Bitcoin by Satoshi, it spread to be adopted by most Blockchain applications. The reason for choosing ECC instead of RSA is that the 2048 key size of RSA provides the same security level of 224bit key size of ECC [11].

Using ECC in Blockchain for securing the identity could provide two properties:

1- The encryption process should be a one-way trap door in which encrypted data is infeasible.
2- Proving of knowing private key without revealing the private key.

Definition

To define ECC more technically, it is all the points in the curve which satisfy the equation $y^2 = x^3 + ax + b$ and for avoiding singularity which is invalid curve (see Figure 11), the condition $4a^3 + 27b^2 \neq 0$ is used. Bitcoin uses the secp256k1 algorithm which defines the parameters of an elliptic curve, and its equation is $y^2 = x^3 + 7$ [12] (see Figure 12).

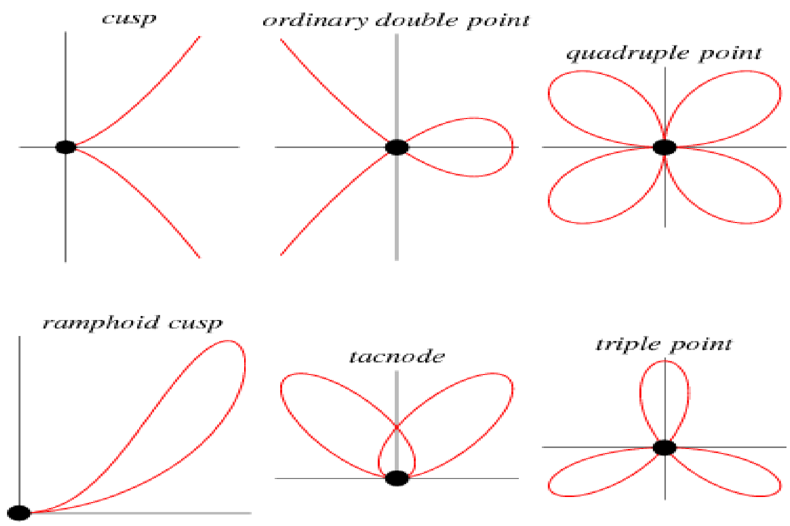

Figure 11 Singularity

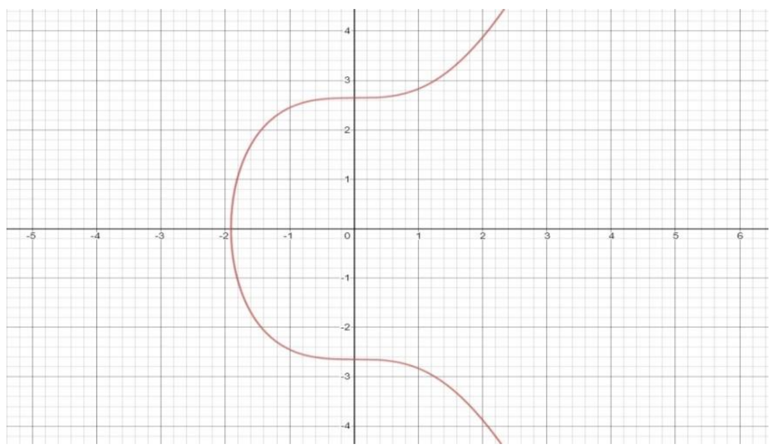

Figure 12 secp256K1 curve

Point addition

Adding two points to get a third point in the curve happened by the following method. Drawing a line between the two points, the third point which intersects with the curve should be the third point. For example, two point p (1,2) + Q(3,4) = R(-3,2). R is the inverse of point -R in x-axis (see Figure 13).

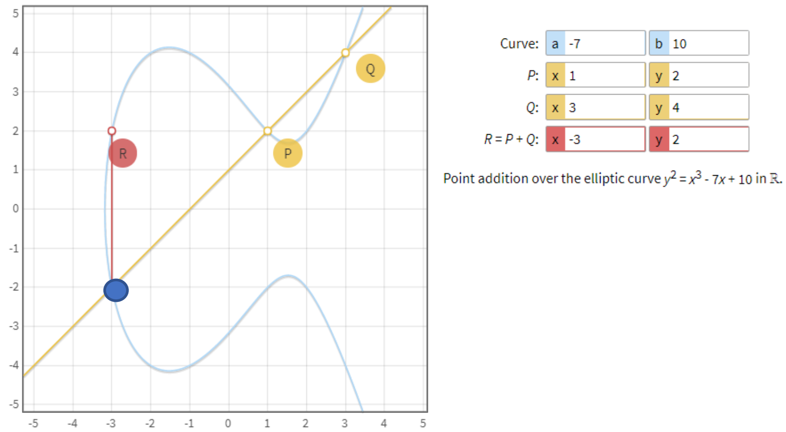

Figure 13 ECC a simple example of two points addition

How point addition works in ECC

To add a point in an Elliptic curve, first, the base point P should be determined in the curve, for example in secp256 the x-coordinate and y-coordinate of point P is predefined as:

x-coordinate

55066263022277343669578718895168534326250603453777594175500187360389116729240

y-coordinate

32670510020758816978083085130507043184471273380659243275938904335757337482424

Then, adding P to itself repeatedly, but how to add P point to itself while there are infinite probabilities lines. In this case, we obtain a tangent line to get the 2P point (see Figure 14)[13].

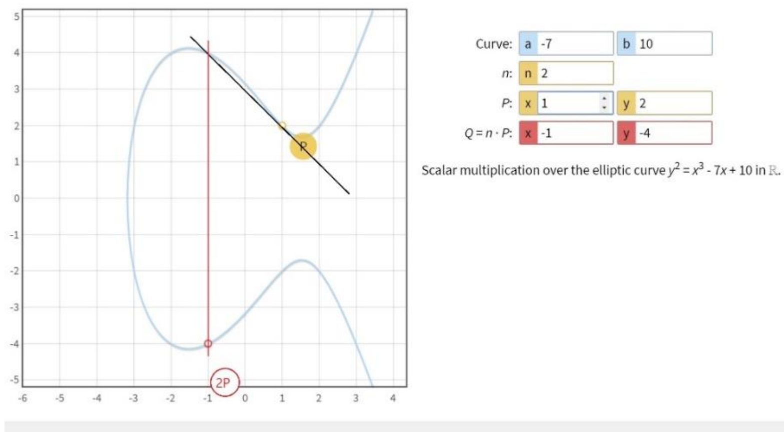

Figure 14 tangent line

Doubling and addition

In this case, it was performed with just two points of addition. However, what if addition points was n times and n for instance was 540. Then it needs 539 additional points ($P_1 + P_2 + P_3 + \ldots\ldots + P_{n-1}$), so this method is not practical. Then for minimizing that, there is a method called doubling and addition where 540 with bits 1000011100 can be converted to power of two:

$$540P = 2^9 * 1 + 2^8 * 0 + 2^7 * 0 + 2^6 * 0 + 2^5 * 0 + 2^4 * 1 + 2^3 * 1 + 2^2 * 1 + 2^1 * 0 + 2^0 * 0$$

$$= 2^9P + 2^4P + 2^3P + 2^2P$$

In this method, 540 was performed just in 9 doubles and 4 additions as is shown in table 1.

Table 1 Doubling and addition for 540P

Addition

doubling

	0	1	2	3	4	5	6	7	8	9
0	2^0									
1	2^1									
2		2^2								
3			$2^3 + 2^2$							
4				$2^4 + 2^3 + 2^2$						
5				2^5						
6				2^6						
7				2^7						
8				2^8						
9					$2^9 + 2^4 + 2^3 + 2^2$					

Finite field and subgroup

In the real situation of an elliptic curve, instead of generating points over the real number R, it will be over a finite field F_p. So, the definition will be as if ECC is all the points over a finite field. In addition, the ECC over F_p keeps the properties of an abelian group [12]. The field is performed in the equation of $y^2 = x^3 + ax + b$ by adding mod P to be like $y^2 = x^3 + ax + b \pmod{P}$. That will make the production of points happen cyclically and shrink the number of points in the field. For example, if an elliptic curve has F_{197} and has order N 216 then based on Lagrange's theorem that n (the order of subgroup) should be a divisor of N. By taking the smallest divisor in which nP = 0, we can find the order of the subgroup. In F_{197} the divisors are 1, 2, 3, 4, 6, 8, 9, 12, 18, 24, 27, 36, 54, 72, 108, 216. $P \neq 0$, $2P \neq 0$ … $8P = 0$. Consequently, the order of the subgroup of the previous example is 8 which produces 0 (see Figure 15).

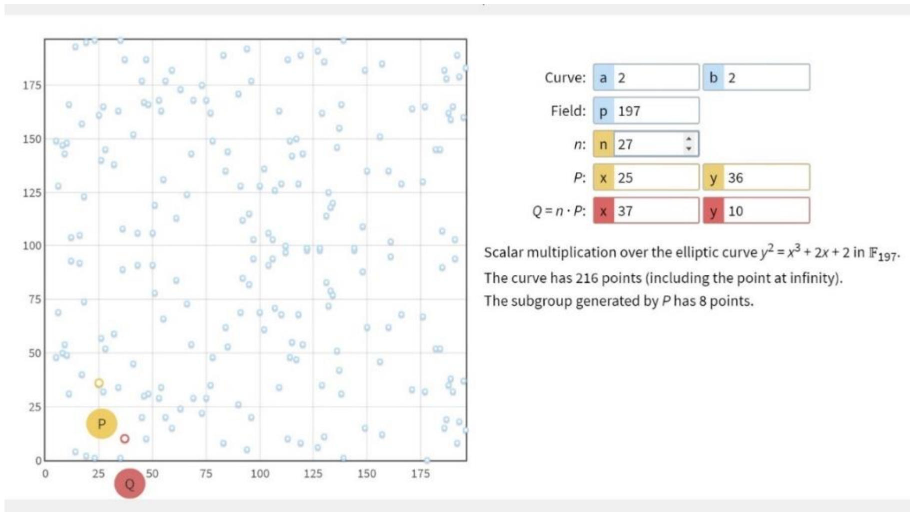

Figure 15 Elliptic curve over F_{197}

Digital signature ECDSA

To sign a message in an Elliptic curve we need Alice (who will sign the message) public key H_A and Alice private key D_A. Alice will sign the hash of the message z, not the message itself (transaction in Blockchain). Now a random integer k from n to n – 1 is chosen. After that p = kG, in which G is the generator point, is calculated. Then r = x_P (mod n), where x_P is x coordinate of the point P, is also calculated. If r = 0 then another k is chosen. The multiplicative inverse of k based on k * k^{-1} mod n = 1 is calculated. Then, s = k^{-1} (z + r D_A) mod n is calculated. If s = 0 then another k is chosen [14].

Lastly, the signed message is sent to Bob. The question here is how Bob could verify the signature?

After receiving the hash message from Alice, Bob will do the following calculation:

1- $U1 = s^{-1} z \bmod n$.
2- $U2 = s^{-1} r \bmod n$.
3- $P = u1G + u2\, H_A$.

If $r = xp \; mod \; n$ then the signature is valid

Bob has to know the public key of Alice H_A, r, and s. The figure shows how the digital signature works (see Figure 16).

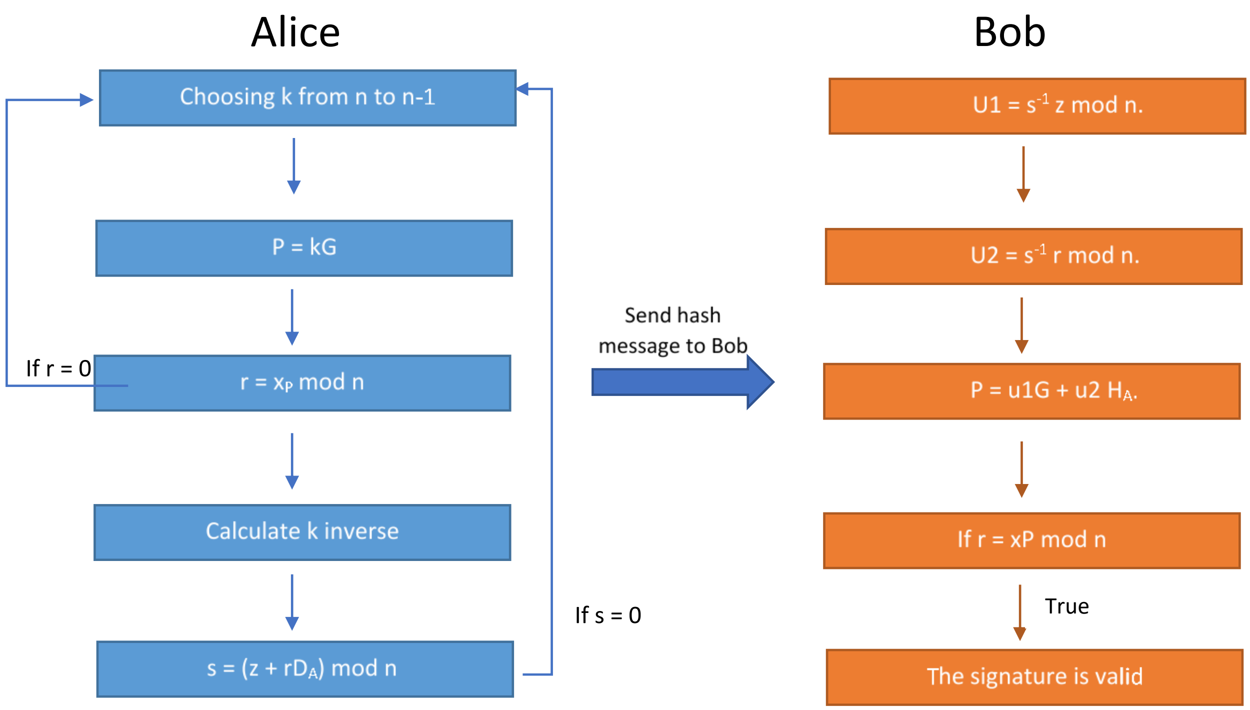

Figure 16 Digital signature ECDSA

Chapter 4 NTRU Cryptography

Introduction

NTRU is a post quantum cryptosystem which is based on a different mathematical problem from ECC. NTRU's name is an abbreviation of N-Th Degree Truncated Polynomial Ring. The objects in NTRU are based on all the truncated polynomial in ring $R = \frac{z[x]}{(x^n - 1)}$ which has degree N-1 and integer coefficient [15].

NTRU was published in 1996 by three mathematicians: Jill Pipher, Jiffrey Hofsein, and Joseph H. Severman. The three inventors of NTRU proclaimed that NTRU creation keys are simple with a high speed and low consumption of memory[16]. NTRU is based on a very difficult problem called closest lattice vector problem (CVP), while in the ECC cryptosystem based on a discrete mathematics problem. As a result, it supposed that there is no polynomial time algorithm that can solve it [15], [17]. As a result, NTRU is more suitable for resisting quantum computers and faster than the other regular cryptosystems.

Parameters and spaces

NTRU uses six parameters, the first three can determine the level of the security in NTRU as is shown in Table 2.

Table 2 Security levels of NTRU

Parameters	N	q	p
Moderate security	167	128	3
Standard security	251	128	3
High security	347	128	3
Very high security	501	256	3

N: polynomial with degree N-1 is better to be prime to acquire more security

q: large prime modulus number for reducing the coefficient

p: small prime modulus number for reducing the coefficient

f: polynomial private key

g: polynomial for generating the public key (should be private)

r: random polynomial for blinding

d: coefficient

L_f: set of polynomials for which private key will be chosen.

$L_{g:}$ set polynomial for which another private key will be chosen.

L_m: set of polynomials for plaintext space and coefficient lie between $-\frac{p-1}{2}$ and $\frac{p-1}{2}$

L_r: set of polynomials where the blinding value will be chosen.

Key generation (Alice)

- Choosing a private polynomial f from set L_f
- Calculate $f * f^{-1} \equiv 1$ (mode p) and $f * f^{-1} \equiv 1$ (mode q). If the inverse f^{-1} does not exist then go back to the first step.
- Calculate public key $h = g*f_q$ (mode q).
- Publish N, h, p, q, L_f, L_g, L_r and L_m to Bob.
- Keep f, f_p private.

Encryption (Bob)

- Put the message m as polynomial set $m = L_m$.
- Choose random r for blinding the message from Lr.
- Calculate $e \equiv p * r * h + m$ (mod q).

Decryption (Alice)

- Calculate a = f*e (mode q).
- Choose the coefficient of a to be set between $^{-q}/_2$ and $^{q}/_2$.
- Calculate $m \equiv f_p * a$ (mode p). Center lifting mod p to get the original message.

Example of encryption and decryption
The parameters will get the values

$N = 7$

$P = 3$

$q = 41$

$d = 2$

Bob

$F(x) = x^6 - x^4 + x^3 + x^2 – 1$

$g(x) = x^6 + x^4 – x^2 -x$

$F_q(x) = f(x)^{-1} \text{ (mod q)} = 8x^6 + 26x^5 + 31x^4 + 21x^3 + 40x^2 + 2x + 37 \text{ (mod 41)}$

$F_p(x) = f(x)^{-1} \text{ (mod p)} = x^6 + 2x^5 + x^3 + x^2 + x + 1 \text{ (mod 3)}$

$h(x) = p * F_q * g \text{ (mod q)} = 19x^6 + 38x^5 + 6x^4 + 32x^3 + 24x^2 + 37x + 8 \text{ (mod 41)}$

sending h and (N, p, q, d) to Alice

$m(x) = -x^5 + x^3 + x^2 - x +1$

$r(x) = x^6 – x^5 + x – 1$

$e(x) = 31x^6 + 19x^5 + 4x^4 + 2x^3 + 40x^2 + 3x + 25 \text{ (mod 41)}$

$a = f*e \text{ (mod q)}$

$a = x^6 + 10x^5 + 33x^4 + 40x^2 + x + 40 \text{ (mod 41)}$

$b = a \text{ (mod p)}$

$b = x^6 + 10x^5 – 8x^4 – x^2 + x – 1 \text{ (mod 3)}$

$c = Fp(x) * b(x)$

$c = 2x^5 + x^3 + x^2 + 2x + 1 \text{ (mod 3)}$

$m = -x^5 + x^3 + x^2 – x + 1$

NTRUSign

NTRUSign is digital signature based on solving approximate closest vector problem. In this section NTRUSign basic operations will be covered as follows [18].

Parameters

N: polynomial with degree N-1 is better to be prime to acquire more security

q: large prime modulus number for reducing the coefficient

d: coefficient

Key Generation

Choosing f, g randomly from $R_q = \frac{\boldsymbol{z_q[x]}}{(\boldsymbol{x^n - 1})}$ which number of ones in f, g are d_f , d_g respectively.

Check if f has inverse, else go back to step1.

Find two small polynomials $F, G \in R$ in which $f * G - g * F = q$

Compute $h = f^{-1} * g \ (modq)$

Public is h. private are f,g

Signing

Assume message m ∈ R

Then the signer will calculate the equation

$$x = \lceil -(1/q) * m * F \rfloor$$

$$y = \lceil (1/q) * m * f \rfloor$$

$$s = x * f + y * F$$

Send (m, s) to the receiver

Verification

check if $\|smodq, (s * h - m)modq\|$ < NormBound, then accept the signature, else the signature is invalid. NormBound is Bound distance between two lattice points

Chapter 5 NTRU and ECC Comparison

Introduction:

In this chapter, a comparison between ECC and NTRU will be studied. First, key sizes between both ECC and NTRU will be shown. Then the previous studies that compared ECC and NTRU in terms of speed will be mentioned.

Key size:

In cryptography key size is specified by bits and can determine the security of the cryptography. The minimum length of the key to be considered secure is 80bit. 128 bits is the most used these days for more security. The table below describes the key sizes of ECC and NTRU and their security level.

Table 3 Key sizes and their level of security

Security level (bits)	NTRU (bits)	ECC (bits)
80	2008	163
112	3033	224
128	3501	256
192	5193	384
256	7690	512

In the table above, the public key size for both ECC and NTRU and their security level in bits can be observed. ECC has less public key size comparing with NTRU, but that does not affect the speed of NTRU [17].

Previous Research:

There are a number of researchers that covered the study of the comparison between ECC and NTRU Cryptosystems for both security and speed.

Miri Jamel, et al [16] proposed a way to substitute the symmetric crypto system which is used in Ad-Hoc protocol with asymmetric cryptosystems. Miri analyzed the performance of the most popular public key cryptosystems to implement with a certificateless scheme for Ad-Hoc Ultra-Wideband Impulse Radio (UWB-IR) networks which are RSA, NTRU and ECC. The comparison was implemented by Java Code and the results were in encryption and decryption.

Encryption and decryption

When comparing encryption and decryption, Miri compared ECC and NTRU Cryptosystems, and it was concluded that NTRU is easier for key generation, faster in encryption (see Figure 17) and decryption (see Figure 18), and more efficient for consuming power [16].

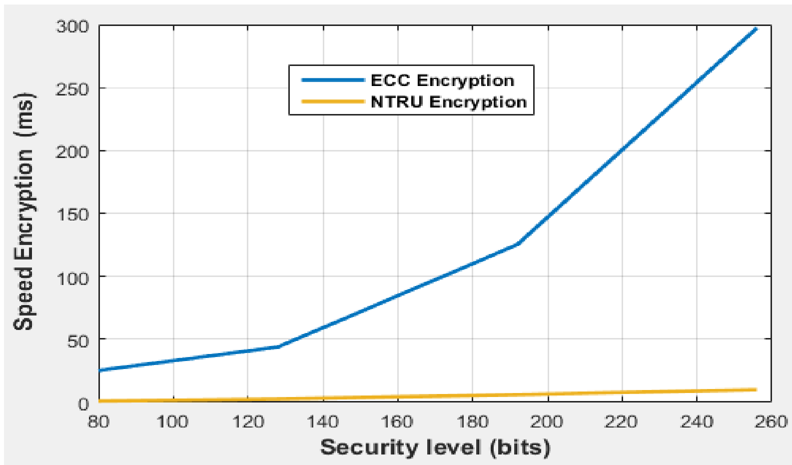

Figure 17 Comparing encryption in ECC and NTRU[16]

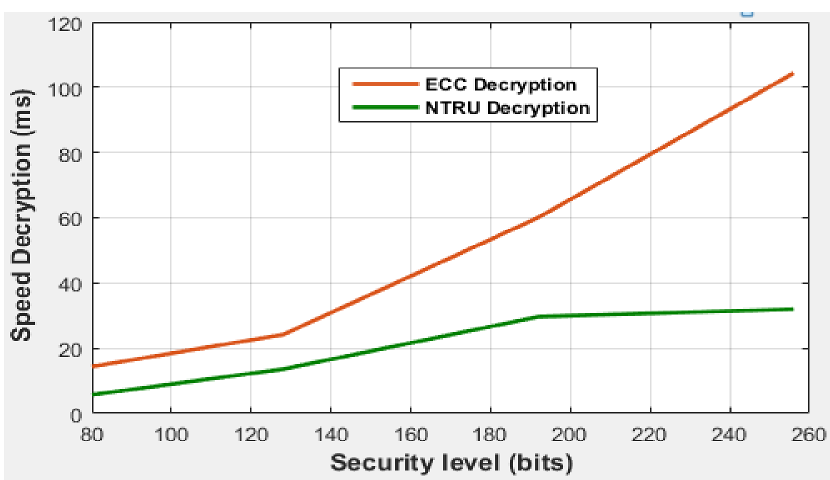

Figure 18 Comparing decryption in ECC and NTRU[16]

Nguyen compared between ECC and NTRU in key size, key generation, encryption and decryption time, the code was compiled on Pentium4 process with 2,4GHz, 1GB RAM, and Windows XP operating system [17]. The table shows the comparison results.

Table 4 ECC and NTRU encryption and decryption [17]

Cryptosystem	Security(bits)	Key Generation(msec)[1]	Encryption(msec)[1]	Decryption(msec)[1]
$NTRU251$	80	75.65	1.68	8.22
$ECC192$	*between* 80 – 112	57.87 – 152.73	37.81 – 116.39	19.15 – 57.68
$NTRU347$	112	114.16	3.11	15.70
$ECC224$	112	234.11 – 367.98	52.52 – 164.50	26.35 – 81.52
$NTRU397$	128	188.92	3.97	20.26
$ECC256$	128	478.22 – 656.63	68.72 – 223.29	35.00 – 111.16
$NTRU491$	160	288.31	5.97	30.96
$NTRU587$	192	412.10	8.42	44.42
$ECC384$	192	947.43 – 1429.11	182.35 – 586.20	90.61 – 290.94
$NTRU787$	256	738.75	14.49	48
$ECC521$	256	2055.04 – 3175.87	423.25 – 1257.56	211.35 – 626.33

Table 4 shows the values of ECC as the minimum and maximum value of all operations that were done.

Nguyen presented the results in graphs as follows. figures 19 to 21 shows NTRU and ECC minimum values in key generation, encryption, and decryption respectively.

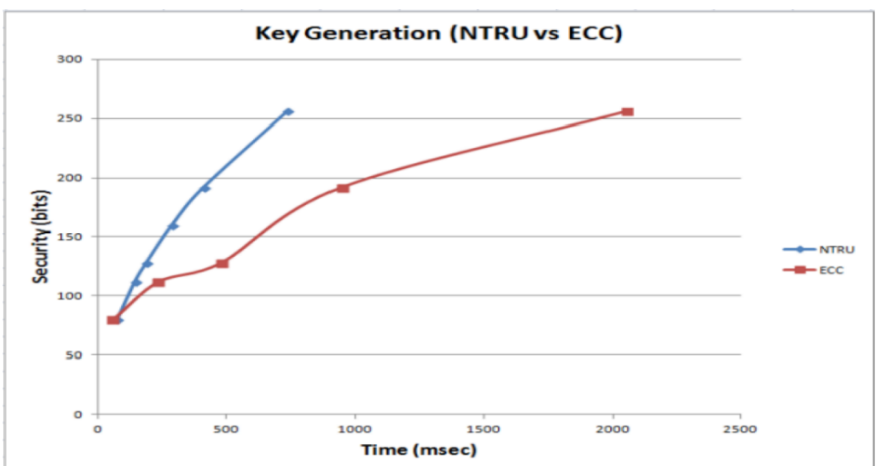

Figure 19 ECC and NTRU key generation [17]

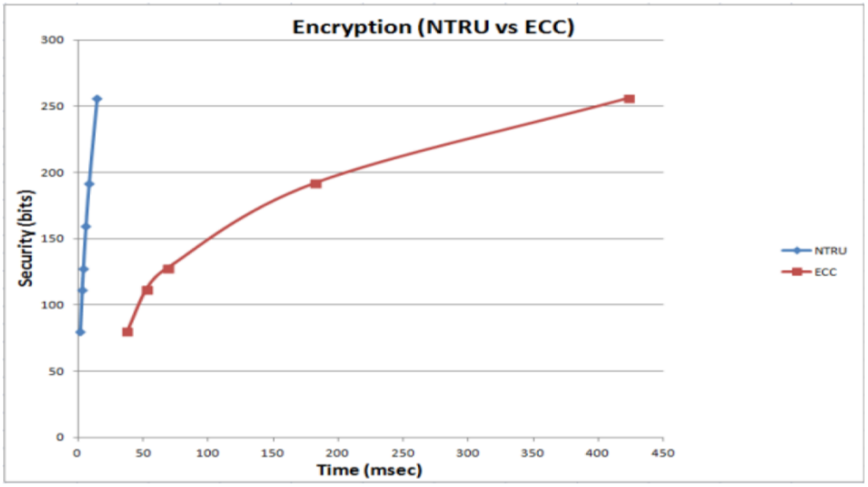

Figure 20 ECC and NTRU encryption time [17]

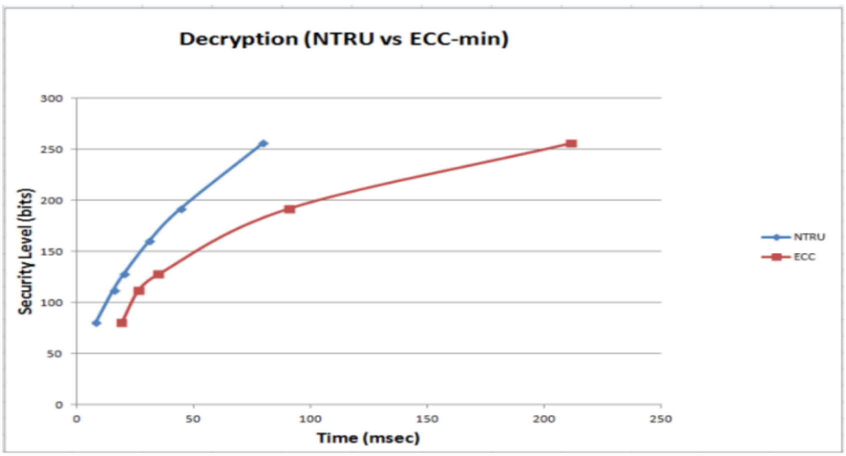

Figure 21 ECC and NTRU decryption time [17]

As a result, we can see that NTRU is much faster than ECC [17].

The NTRUSign original paper Jeffry Hoffestan, et al [19] explained NTRUSign in detail and compared NTRUSign with ECDSA in both signing the message and verifying by using Pentium Machine 800 MHs[19]. The results of this comparison is described in table 5.

Table 5 Comparison of NTRUSign, ECDSA, RSA [19]

	NTRUSign-251	ECDSA-163	RSA-1024
Keygen (µs)	180,000	1424	500,000
Sign (µs)	500	1424	9090
Verify (µs)	303	2183	781

Juliet [20] proclaimed that because of the evolution of computer technologies toward the quantum, the need for using alternative post quantum resistance like NTRU is necessary. In the research, NTRU was stated to be a better alternative for RSA and ECC as a result of being faster and lower consumption. The comparison of performance between NTRU, ECC, and RSA was applied on server 800MHz Pentium III processor Using C code, and portable device with the following results (see Table 6).

Table 6 Comparison of NTRU, ECC and RSA on a server and on a constrained device [20]

		Speed on server			Speed on portable device		
Function	Units of measurement	NTRU 251	ECC 163	RSA 1024	NTRU 251	ECC 163	RSA 1024
Encryption	Block/sec	22727	48	1280	21	0.4	0.5
Decryption	Block/sec	10869	55	110	12	1.3	0.036

The result shows that NTRU outperformed ECC in both the server and portable device in encryption and decryption.

Seo [21] compared RSA, ECC and two post quantum cryptosystems NTRU and Lizard. The comparison was in the speed of three parts: encryption, decryption, and key generation. The implementation was written in C Code and tested on IMAC with i7 CPU 3,7GHz. The security of each cryptosystem is around 128 bits. The results were implemented thousands of times for accuracy and are displayed in Table 7 and Figure 22.

Table 7 Encryption and decryption speeds of various cryptosystems [21]

Encryption Scheme	Key Generation Time (milliseconds)	Encryption Time (milliseconds)	Decryption Time (milliseconds)
RSA	275.477	0.782	28.118
ECC	8.184	6.5	1.5
NTRU	3.829	0.438	0.826
Lizard	15.521	0.009	0.009

From the above-mentioned data, it can be noticed that NTRU is faster than ECC in encryption, decryption, and key generation.

Conclusion

Through the previous works, there is an agreement that NTRU has superior performance and better efficiency. In addition, using NTRU postquantum cryptosystems instead of ECC can enhance the capability of Blockchain.

Chapter 6: Accelerating propagation delay by using NTRU verification process

Introduction

As previously mentioned, propagation delay is the main concept that is responsible for issues like inconsistency, double spending, and majority attacks. Our focus here in this chapter is to reduce the propagation delay by focusing on reducing verification time. And that will be by using NTRUSign instead of ECDSA.

Proposed method

As mentioned in Chapter 5, the NTRUSign is more effective and powerful than ECDSA. The proposed method depends on using NTRUSign to verify transactions rather than using ECDSA. In this case, the verification time will be faster and thereby accelerate propagation time between the nodes.

The model

The code was built by java code which simulates a network that contains several nodes connected with each other. Figures 23-26 shows the structure of the network. The red nodes represent that the nodes received the message but not verify it yet.

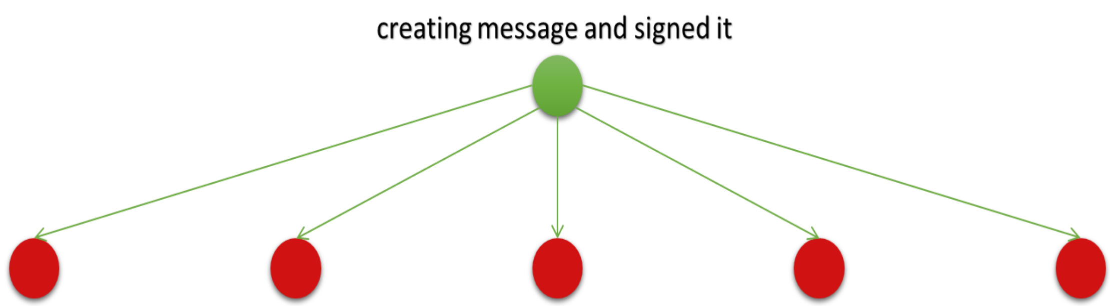

Figure 22 The structure of the proposed method

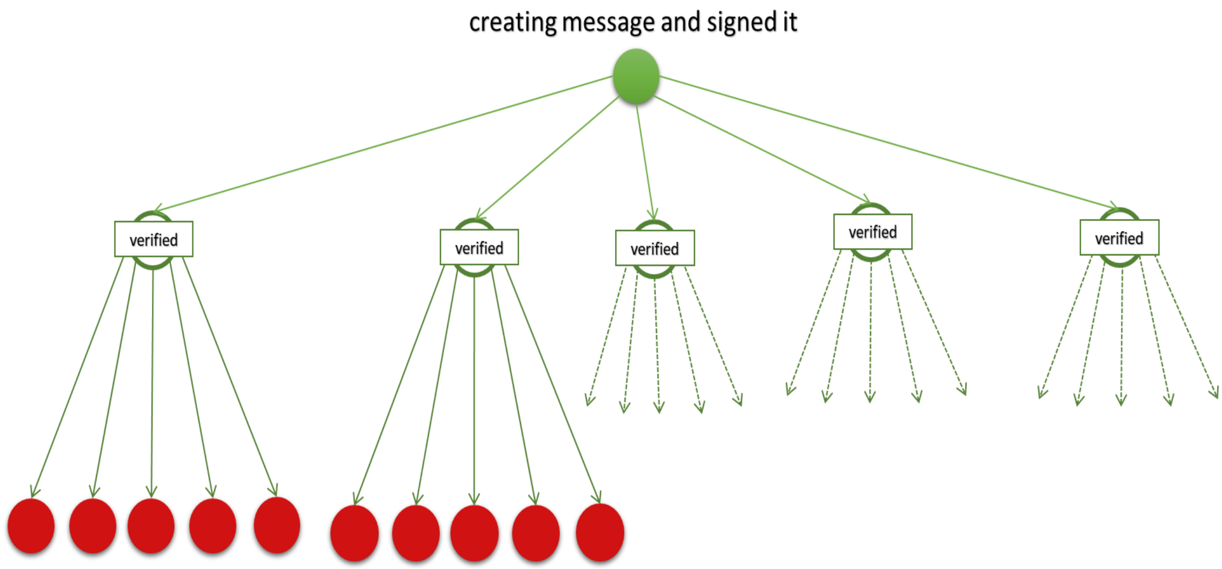

Figure 23 The structure of the proposed method

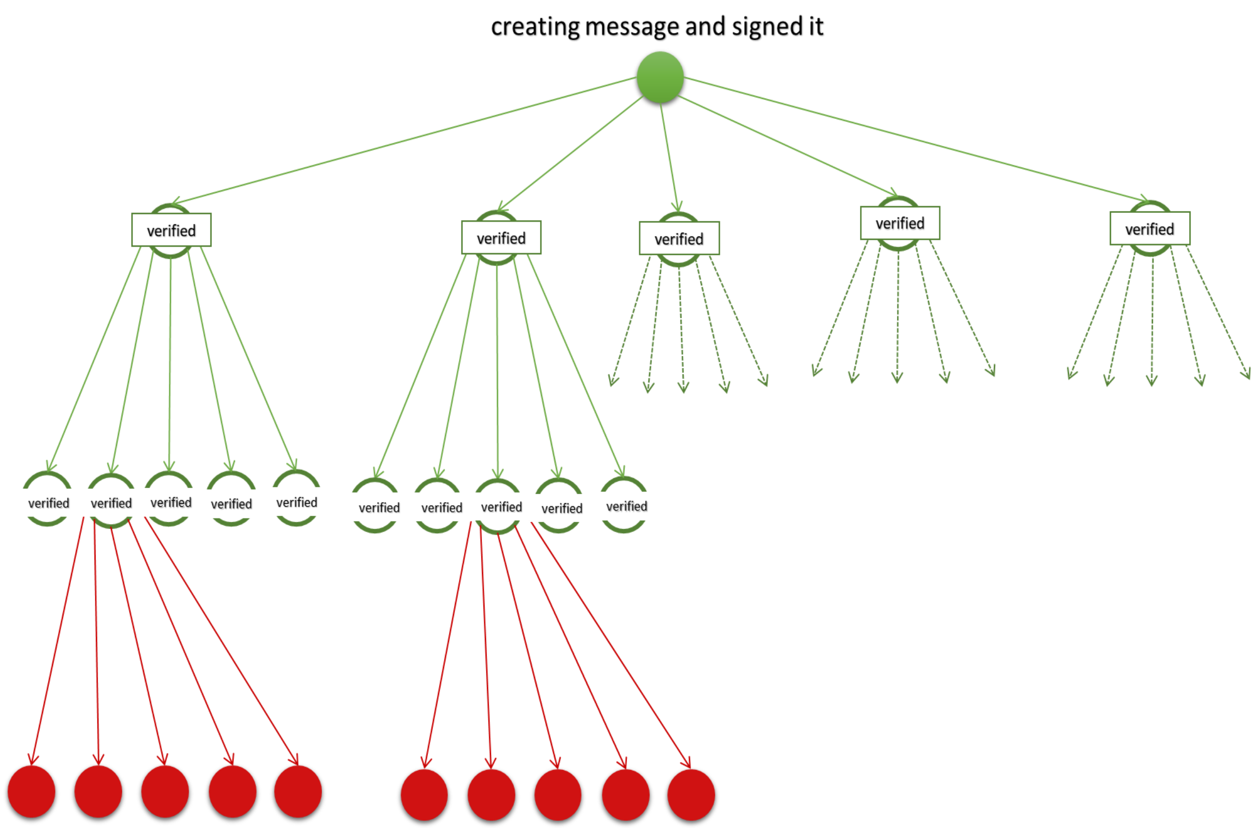

Figure 24 The structure of the proposed method

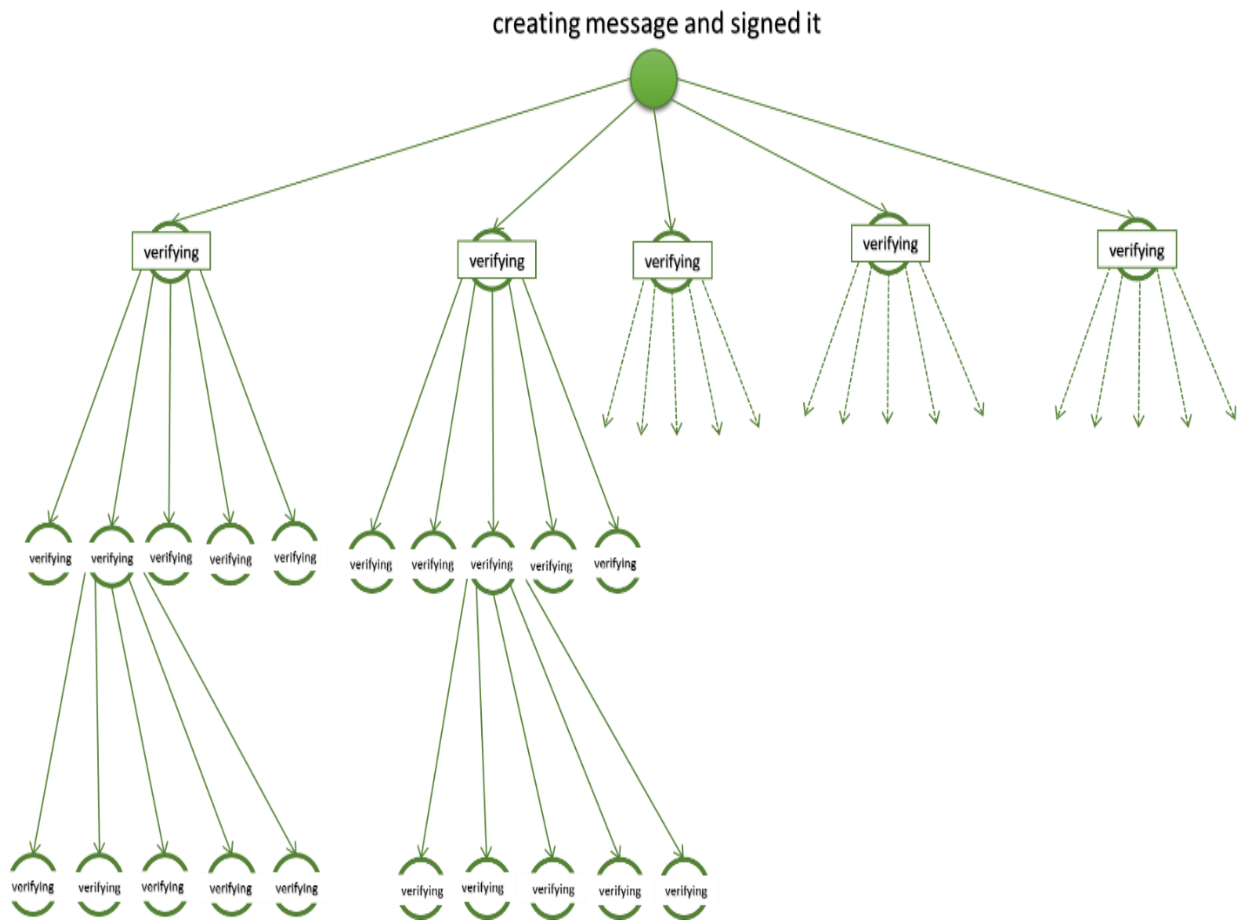

Figure 25 The structure of the proposed method

In Figures 23-26, one of the nodes will create a message and sign it at first with ECDSA and propagate it to all the nodes in the network and each node will verify the message. The time will be taken from the start of the verification process until the end of verification. After that, the same node will create another message and sign it with NTRUSign. Then it will be propagated to all the nodes in the network and each node will verify it. The times for NTRUSign also will be taken from the start of the verification process until the end of the verification.

Implementation

In the simulator, we compared both the digital signature with a different number of nodes and a different number of linked nodes. The code was implemented in java by using Tbuktu NTRUSign [21] code and secp256k1 from BouncyCastle java library. The implementation was based on windows10 64bit and processor intel core i7 with RAM 12GB. The table below shows the different scenarios of testing.

Table 8 The five scenarios of the proposed method

First scenario				
Number of linked nodes	Total number of nodes in the network			
5	100	1000	10,000	100,000
Second scenario				
Number of linked nodes	Total number of nodes in the network			
10	100	1000	10,000	100,000
third scenario				
Number of linked nodes	Total number of nodes in the network			
100	1000	10000	100,000	1000,000
Fourth scenario				
Number of linked nodes	Total number of nodes in the network			
1000	10000	100,000	1000,000	
Fifth scenario				
Number of linked nodes	Total number of nodes in the network			
10,000	100,000	1000,000		

The results

Each phase was implemented separately, the results for each scenario was put in separate tables. Furthermore, the test was implemented 10 times for each number of nodes. Then the average time was taken for the verification process in milliseconds. The five phases in Table 8 have been ordered based on the number of linked nodes which in the first phase will be 5 linked nodes, second 10 linked nodes, third 100 linked nodes, fourth 1000 nodes, and lastly 10000 linked nodes.

First Scenario

Table 9 First scenario, 5 Linked Nodes

	Number of nodes							
5 linked nodes	100		1000		10,000		100,000	
	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA
Minimum(ms)	23	44	231	370	2368	3606	23773	33663
Maximum(ms)	128	144	575	777	2731	5033	24851	37493
Total avg(ms)	53	69	280	520	2447	3787	24070	35008

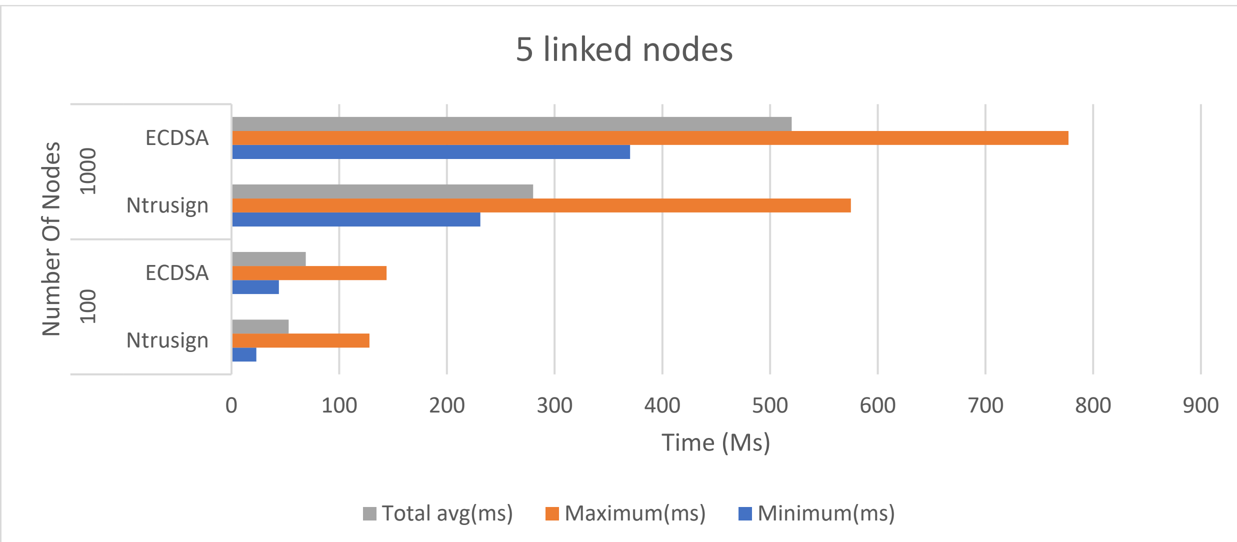

Figure 26 NTRU and ECDSA comparison time with 5 linked nodes, 100 and 1000 Total nodes

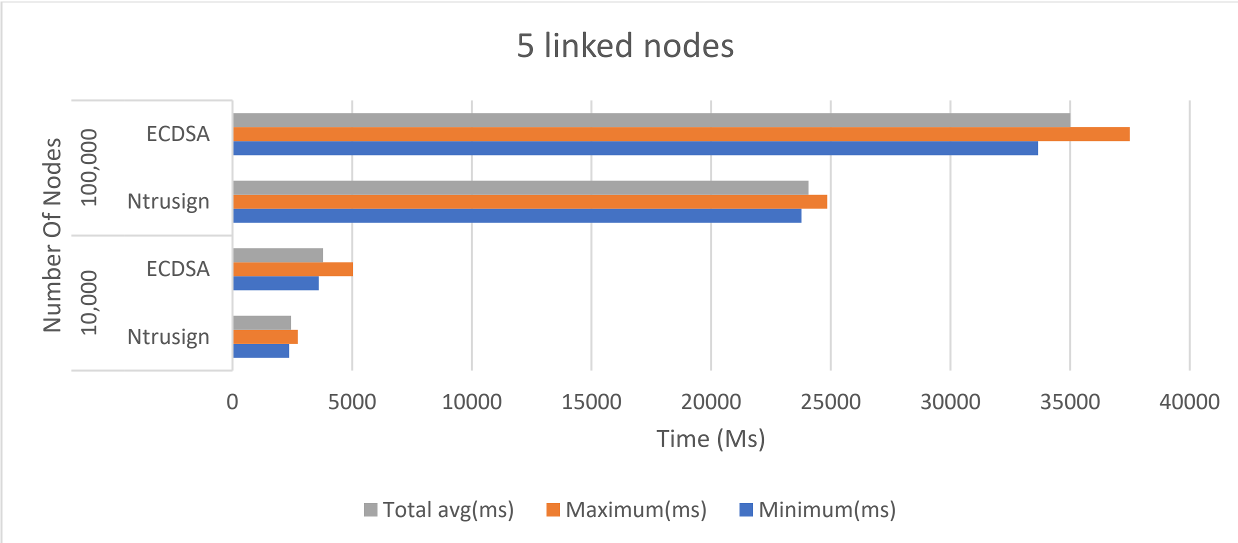

Figure 27 NTRU and ECDSA comparison time with 5 linked nodes, 100 and 1000 Total nodes

Second Scenario

Table 10 Second scenario, 10 Linked Nodes

10 linked nodes	Number of nodes							
	100		1000		10,000		100,000	
	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA
Minimum(ms)	14	27	204	309	2435	3001	22727	33608
Maximum(ms)	116	98	511	599	2790	4175	23061	35644
Total avg(ms)	54	65	243	403	2496	3157	22859	34726

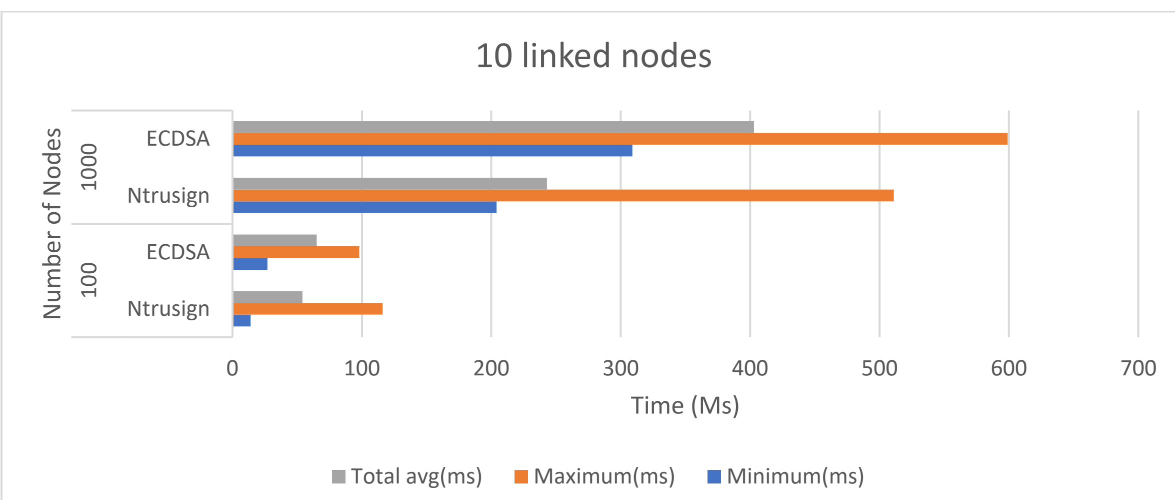

Figure 28 NTRU and ECDSA comparison time with 10 linked nodes, 100 and 1000 Total nodes

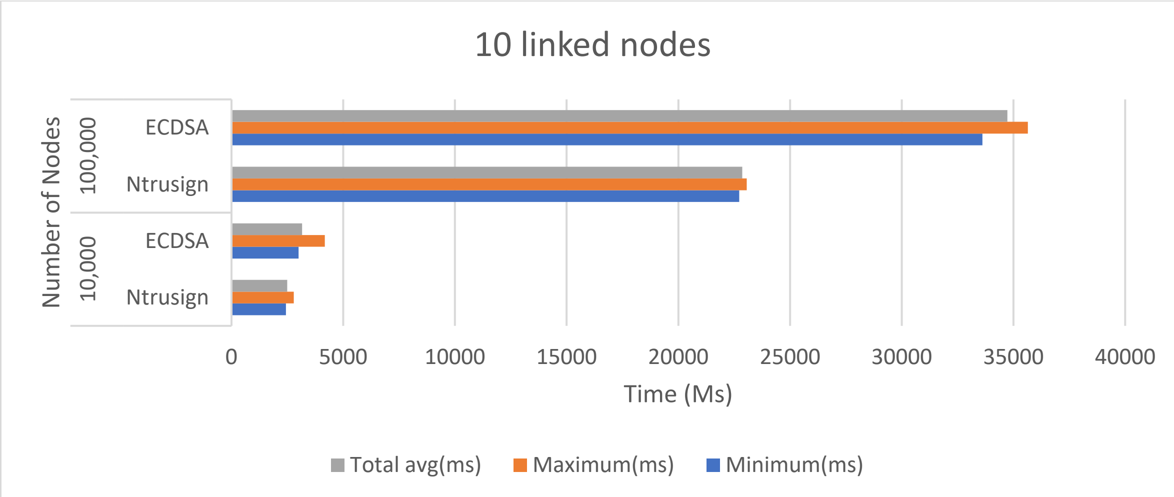

Figure 29 NTRU and ECDSA comparison time with 10 linked nodes, 10,000 and 100,000 Total nodes

Third Scenario

Table 11 Third scenario, 100 Linked Nodes

100 linked nodes	Number of nodes							
	1000		10,000		100,000		1,000,000	
	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA	Ntrusign	ECDSA
Minimum(ms)	204	313	2388	2906	22218	32203	214391	324415
Maximum(ms)	537	615	2878	4091	23003	35411	217211	563841
Total avg(ms)	247	413	2466	3089	22504	33556	215451	362226

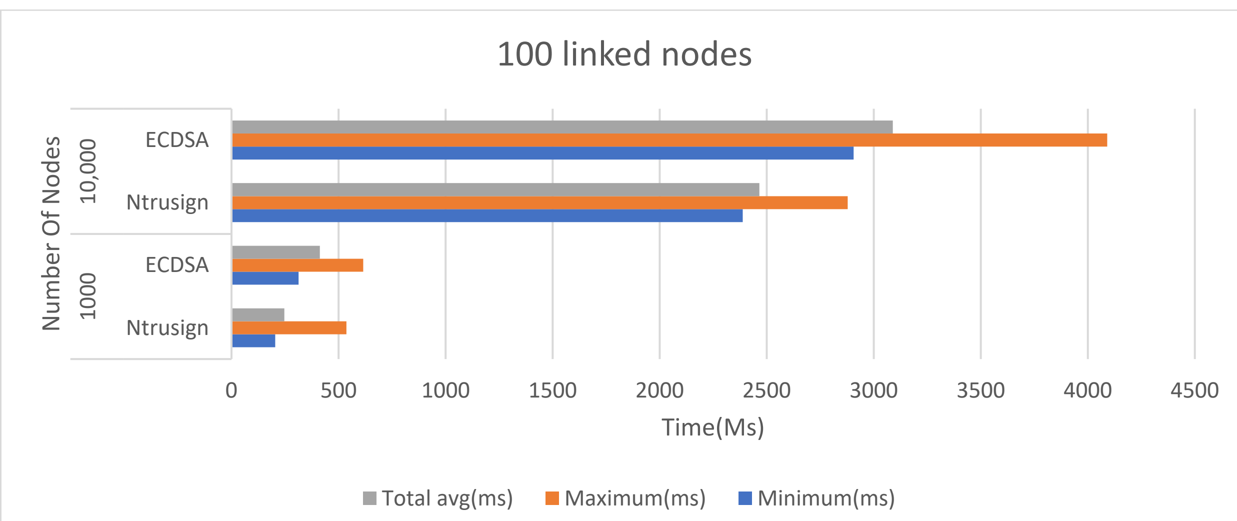

Figure 30 NTRU and ECDSA comparison time with 100 linked nodes, 1000 and 10,000 Total nodes

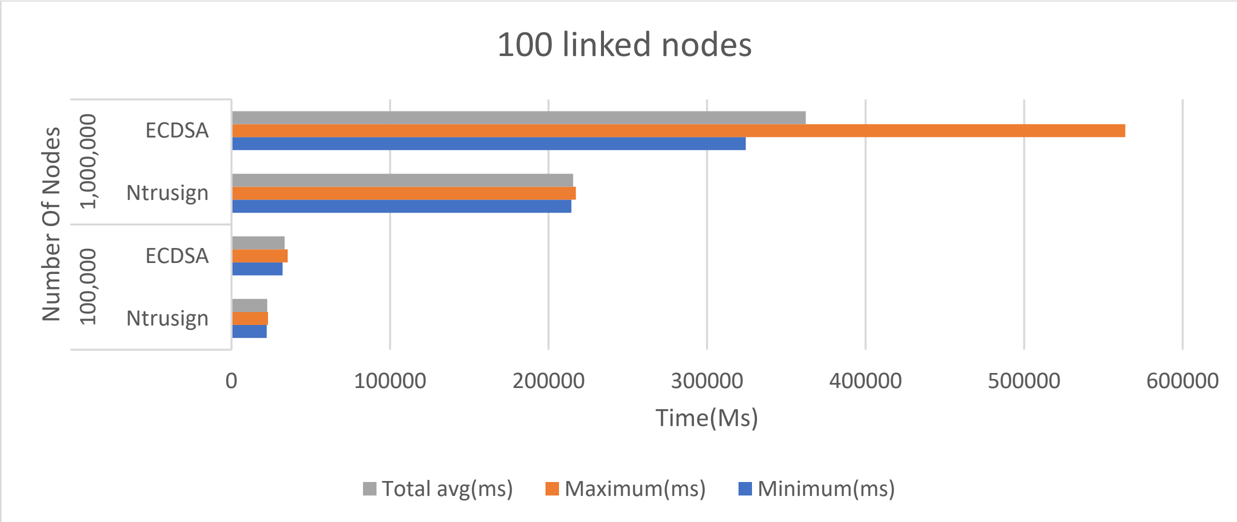

Figure 31 NTRU and ECDSA comparison time with 100 linked nodes, 100,000 and 1000,000 Total nodes

Fourth Scenario

Table 12 Fourth scenario, 1000 Linked Nodes

	Number of nodes					
1000 linked nodes	10,000		100,000		1,000,000	
	Ntrusign(ms)	Ecdsa(ms)	Ntrusign(ms)	Ecdsa(ms)	Ntrusign(ms)	Ecdsa(ms)
Minimum	2268	3418	22994	36058	205251	300811
Maximum	2651	4718	23848	35266	208158	312789
Total avg	2366	3593	23233	35132	206597	303716

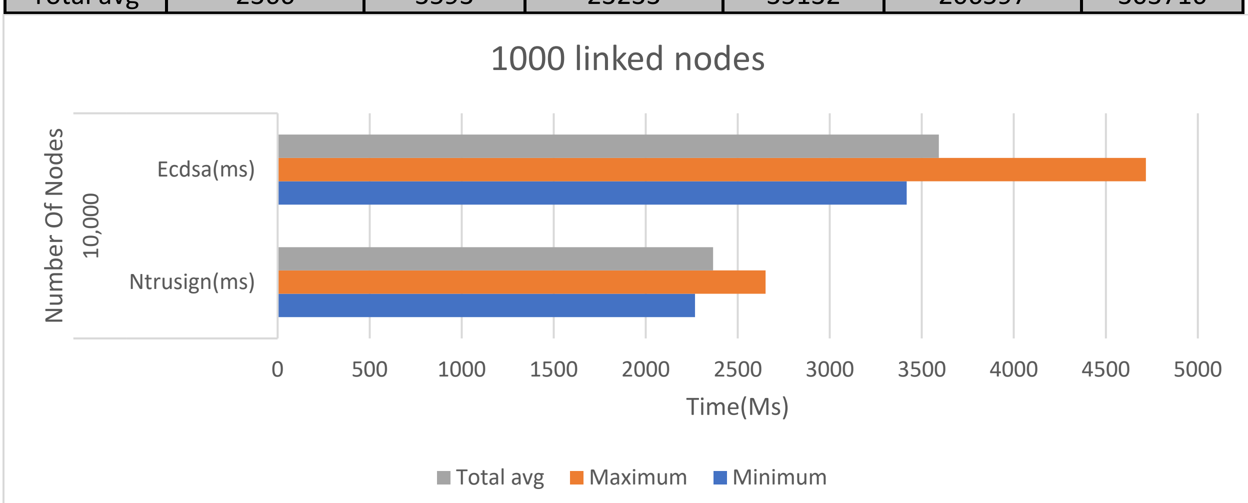

Figure 32 NTRU and ECDSA comparison time with 1000 linked nodes, 10,000 Total nodes

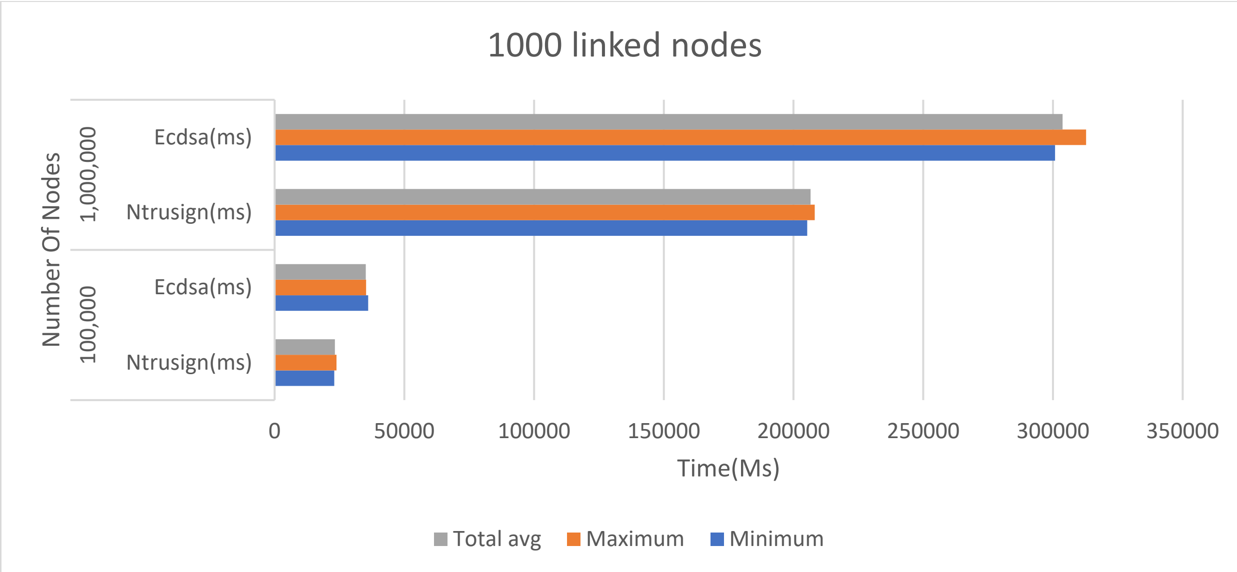

Figure 33 NTRU and ECDSA comparison time with 1000 linked nodes, 100,000 and 1000,000 Total nodes

Fifth Scenario

Table 13 Fifth scenario, 10000 Linked Nodes

10,000 linked nodes	Number of nodes			
	100,000		1,000,000	
	Ntrusign	ECDSA	Ntrusign	ECDSA
Minimum(ms)	21869	31343	207249	304581
Maximum(ms)	22341	32436	209944	320736
Total avg(ms)	22055	31683	208219	311257

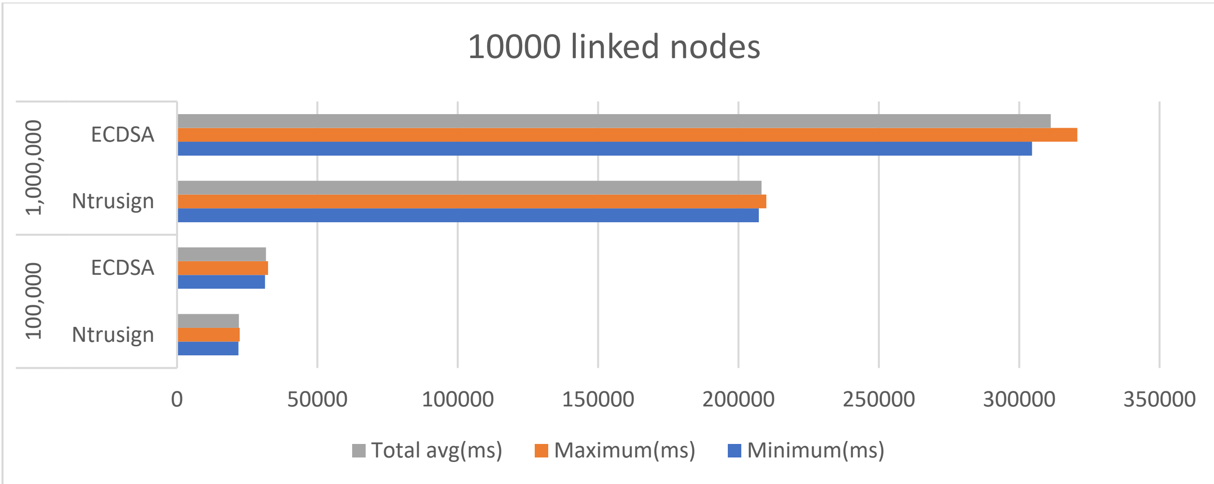

Figure 34 NTRU and ECDSA comparison time with 10,000 linked nodes, 100,000 and 1000,000 Total nodes

Based on the five scenarios, it can be observed that NTRUSign outperformed ECDSA by 32.06% on average after taking the percentage of all the processes.

Conclusion and future work

In this thesis, as mentioned in chapter 3, the propagation delay in Blockchain networks can lead to several vital security issues like double spending attacks, majority attacks, or forking. The proposed method is presented to change the recent ECDSA digital signature to a NTRUSign digital signature with the goal of speeding up the verification time and thereby minimizing propagation delay. By doing this, most of those attacks can be avoided. The result of substituting ECDSA with NTRUSign was satisfying because it shows an enhancement in the speed of the verification process.

In future work, the protection of blockchain network from quantum computer attacks will be focused on and increasing level of security in NTRU.

References

1. “A Cypherpunk’s Manifesto.” [Online]. Available: https://www.activism.net/cypherpunk/manifesto.html.
2. “(weidai).” [Online]. Available: http://www.weidai.com/bmoney.txt. [Accessed: 07-Mar-2020].
3. “What Is Bit Gold? The Brainchild of Blockchain Pioneer Nick Szabo - CoinCentral.” [Online]. Available: https://coincentral.com/what-is-bit-gold-the-brainchild-of-blockchain-pioneer-nick-szabo/.
4. Nakamoto, Satoshi, and A. Bitcoin. "A peer-to-peer electronic cash system." *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf* (2008).
5. *BLOCKCHAIN DECENTRALIZED TRUST*. (Book)
6. “What is Ethereum? | Ethereum.org.” [Online]. Available: https://ethereum.org/what-is-ethereum/. [Accessed: 07-Mar-2020].
7. Bitcoin PROGRAMMING THE OPEN BLOCKCHAIN.” (Book)
8. Li, Xiaoqi, et al. "A survey on the security of blockchain systems." *Future Generation Computer Systems* (2017).
9. Moustapha, B. A. "The effect of propagation delay on the dynamic evolution of the Bitcoin blockchain." *Digital Communications and Networks* (2019).
10. Decker, Christian, and Roger Wattenhofer. "Information propagation in the bitcoin network." *IEEE P2P 2013 Proceedings*. IEEE, 2013.
11. “What is the math behind elliptic curve cryptography?” [Online]. Available: https://hackernoon.com/what-is-the-math-behind-elliptic-curve-cryptography-f61b25253da3.
12. Hankerson, Darrel, Alfred J. Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.
13. “Elliptic Curve Cryptography: a gentle introduction - Andrea Corbellini.” [Online]. Available: https://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/..
14. Hoffstein, Jeffrey, Jill Pipher, and Joseph H. Silverman. "NTRU: A ring-based public key cryptosystem." *International Algorithmic Number Theory Symposium*. Springer, Berlin, Heidelberg, 1998.
15. Nitaj, Abderrahmane. "The Mathematics of the NTRU Public Key Cryptosystem." (2015).
16. Miri, Jamel, Bechir Nsiri, and Ridha Bouallegue. "Certificateless Scheme Based NTRU Cryptosystem for Ad-Hoc UWB-IR Network." *International Journal of Wireless & Mobile Networks (IJWMN) Vol* 9 (2017).
17. Nguyen, Hien Ba. *An Overview On The Ntru Cryptographic System*. Diss. Sciences, 2014.
18. Li, Daofeng, et al. "A New Self-Certified Signature Scheme Based on NTRUS ing for Smart Mobile Communications." *Wireless Personal Communications* 96.3 (2017): 4263-4278.
19. Hoffstein, Jeffrey, et al. "NTRUSIGN: Digital signatures using the NTRU lattice." *Cryptographers’ Track at the RSA Conference*. Springer, Berlin, Heidelberg, 2003.
20. Gaithuru, Juliet N., and Majid Bakhtiari. "Insight into the operation of NTRU and a comparative study of NTRU, RSA and ECC public key cryptosystems." *2014 8th. Malaysian Software Engineering Conference (MySEC)*. IEEE, 2014.

21. Seo, William Yunsoo. "Comparing RSA ECC and post quantum cryptography." *J. Math. Anal. Appl.* 10 (2018): 19-33.
22. "NTRU - Browse /NTRU 1.2 at SourceForge.net." [Online]. Available: https://sourceforge.net/projects/ntru/files/NTRU%201.2/. [Accessed: 25-Apr-2020].
23. Bi, Wei, Huawei Yang, and Maolin Zheng. "An accelerated method for message propagation in blockchain networks." *arXiv preprint arXiv:1809.00455* (2018).
24. Sudhan, Amool, and Manisha J. Nene. "Peer Selection Techniques for Enhanced Transaction Propagation in Bitcoin Peer-to-Peer Network." *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, 2018.
25. Bamert, Tobias, et al. "Have a snack, pay with Bitcoins." *IEEE P2P 2013 Proceedings*. IEEE, 2013.
26. Marçal, João, Luís Rodrigues, and Miguel Matos. "Adaptive information dissemination in the Bitcoin network." *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. 2019.
27. Eyal, Ittay, and Emin Gün Sirer. "Majority is not enough: Bitcoin mining is vulnerable." *International conference on financial cryptography and data security*. Springer, Berlin, Heidelberg, 2014.
28. Vallois, Valentin, and Fouad Amine Guenane. "Bitcoin transaction: From the creation to validation, a protocol overview." *2017 1st Cyber Security in Networking Conference (CSNet)*. IEEE, 2017.
29. Li, Xiaoqi, et al. "A survey on the security of blockchain systems." *Future Generation Computer Systems* (2017)..
30. Yli-Huumo, Jesse, et al. "Where is current research on blockchain technology?—a systematic review." *PloS one* 11.10 (2016): e0163477.
31. Zheng, Zibin, et al. "Blockchain challenges and opportunities: A survey." *International Journal of Web and Grid Services* 14.4 (2018): 352-375.
32. "This tutorial describes how the NTRU Public Key Cryptosystem (PKCS) works," 2014.
33. D. Drescher, *Blockchain basics: A non-technical introduction in 25 steps*. Apress Media LLC, 2017.
34. Sompolinsky, Yonatan, and Aviv Zohar. "Accelerating Bitcoin's Transaction Processing. Fast Money Grows on Trees, Not Chains." *IACR Cryptology ePrint Archive* 2013.881 (2013).
35. Karame, Ghassan O., Elli Androulaki, and Srdjan Capkun. "Double-spending fast payments in bitcoin." *Proceedings of the 2012 ACM conference on Computer and communications security*. 2012.
36. Kan, Jia, et al. "Boost Blockchain Broadcast Propagation with Tree Routing." *International Conference on Smart Blockchain*. Springer, Cham, 2018.